frontiers | Frontiers in Energy Research

TYPE Brief Research Report
PUBLISHED 12 July 2024
DOI 10.3389/fenrg.2024.1424663

OPEN ACCESS

EDITED BY
Zhi-Wei Liu,
Huazhong University of Science and Technology, China

REVIEWED BY
Ting Wu,
Harbin Institute of Technology, China
Caishan Guo,
South China University of Technology, China
Guibin Wang,
Shenzhen University, China

*CORRESPONDENCE
Zewen Li,
lizewenfjgs@outlook.com

RECEIVED 28 April 2024
ACCEPTED 25 June 2024
PUBLISHED 12 July 2024

CITATION
Fan Y, Wu H, Li Z, Lin J, Li L, Huang X, Chen W and Chen B (2024), A flexible orchestration of lightweight AI for edge computing in low-voltage distribution network.
Front. Energy Res. 12:1424663.
doi: 10.3389/fenrg.2024.1424663

A flexible orchestration of lightweight AI for edge computing in low-voltage distribution network

Yuanliang Fan, Han Wu, Zewen Li*, Jianli Lin, Lingfei Li, Xinghua Huang, Weiming Chen and Beibei Chen

Electric Power Research Institute of Fujian Electric Power Co., Ltd., Fuzhou, China

Recent years, the tremendous number of distributed energy resources, electric vehicles are integrated into the Low-Voltage Distribution Network (LVDN), large amount of data are generated by edge devices in LVDN. The cloud data centers are unable to process these data timely and accurately, making it impossible to meet the demand for fine-grained control of LVDN. To solve the above problems, this paper proposes a flexible orchestration of lightweight artificial intelligence (AI) for edge computing in LVDN. Firstly, the application requirements of LVDN are analysed through feature extraction of its historical data, and a lightweight AI library is constructed to meet its requirements. Secondly, based on the multi-factor priority, a flexible orchestration model is established, to allow the lightweight AI embedded in the edge devices of the LVDN. Finally, the particle swarm optimization algorithm is used to provide the best solution. The simulation results show that the method proposed in this paper can support the deployment of AI at the edge. It can significantly improve the utilization of edge computing resources, and reduce the pressure of cloud computing and the time of application processes.

KEYWORDS

lightweight AI, edge computing, low-voltage distribution network, cloud-edge computing, particle swarm optimization algorithm

1 Introduction

The Low-Voltage Distribution Network (LVDN) is an important facility for delivering electricity to end-users. Currently, distributed energy resources and electric vehicles which are integrated into the LVDN, bring great challenges to the operation and regulation of the LVDN. The traditional centralised cloud-based scheduling model is no longer able to process the massive data generated by edge devices efficiently (Wang and Peng, 2020; Li et al., 2022).

Edge computing is an effective solution to address the problem of large number of terminal access and massive data processing. Edge computing uses intelligent processing at the edge of the LVDN, such as intelligent fusion terminals, IoT switches, numerous sensors and other edge devices, to integrate network, computing, storage, and information technology to provide services at the edge of the LVDN (Fu et al., 2022; Mudassar et al., 2022). By processing at the edge of the network, it can make the business processes of the LVDN transfer from the cloud-based power system to the edge side, reduce the pressure on the core nodes of the cloud network, and have significant advantages

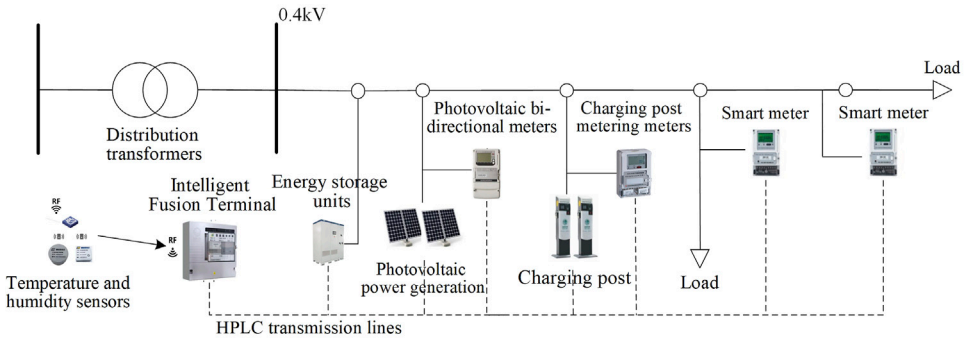

FIGURE 1
Distribution network topology map and intelligent fusion terminal application scenario.

in improving the response speed and optimising the efficiency of the business services (Chen et al., 2019; Han et al., 2022).

At present, numerous studies investigate the utilization of edge computing in LVDN. These research efforts encompass various areas such as distributed power fault detection, distribution network management and the energy consumption of edge computing. The first is to detect the distributed power fault (Huo et al., 2020; Peng et al., 2021; Cai et al., 2023). Use edge computing to detect the distributed power distribution fault, enabling timely sensing and real-time response to faults with LVDN. The second is to enhance distribution network management (Wang et al., 2023; Yue et al., 2023; Zhong et al., 2023). Design the orchestration mechanism of edge computing to speed up the response time of power dispatching and to deal with voltage alert problems (Chamola et al., 2020). Use electric vehicles as a reactive power resource to bolster the reliability of LVDN. The third is the energy consumption of edge computing (Chen et al., 2022; Perin et al., 2022). Use distributed online resource allocation and load management to reduce energy costs and energy consumption. However, these studies have not explored the application of AI models on certain devices like power distribution fusion terminals, which lack adequate computational resources.

Consequently, this paper proposes a flexible orchestration of lightweight AI for edge computing in LVDN. The challenges associated with model deployment and operation in resource-constrained environments can be addressed at the edge end of LVDN. Firstly, the application requirements of LVDN are analysed through feature extraction of its historical data, and a lightweight AI library is constructed to meet its requirements. Secondly, based on the multi-factor priority, a flexible orchestration model is established, to allow the lightweight AI embedded in the edge devices of LVDN. This flexibility enables dynamic allocation and management of computational resources to ensure that the AI is efficiently allocated and executed on edge devices in LVDN. Finally, the particle swarm optimization algorithm is used to provide the best solution. The simulation results show that the method proposed in this paper can support the deployment of AI at the edge. It can significantly improve the utilization of edge computing resources, and reduce the pressure of cloud computing and the time of application processes.

2 Application scenarios analysis and lightweight AI library construction for LVDN

2.1 Feature extraction and application scenarios identification of LVDN

The first step involves creating a scene feature set which employ the equipment ledger and historical operation data for LVDN, Factors such as the scale of the LVDN, the percentage of distributed energy resources, and the configuration of energy storage systems are considered, as shown in Figure 1.

2.1.1 Feature extraction for LVDN

Firstly, historical operation data uploaded from LVDN is collected. As well as collecting real-time data from sensors, monitoring devices. A feature set appropriate to the distribution station is formed, which can be categorized into statistical features and curve features. The statistical features include numerical data such as the proportion of PV access, the capacity of energy storage, the average daily load, the peak-valley load difference. The curve features include sequential data such as distributed PV output curves, distribution transformer load curves, and charge and discharge curves of storage or charging piles. For the ith station scenario, its statistical-type features and curve-type features are as follows:

$$X_i = \left[x_1^i, x_2^i, \cdots, x_j^i, \cdots x_M^i\right] \tag{1}$$

$$F_i = \begin{bmatrix} f_{1,1}^i & f_{1,2}^i & \cdots & f_{1,k}^i & \cdots & f_{1,L}^i \\ f_{2,1}^i & f_{2,2}^i & \cdots & f_{2,k}^i & \cdots & f_{2,L}^i \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ f_{j,1}^i & f_{j,2}^i & \cdots & f_{j,k}^i & \cdots & f_{j,L}^i \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ f_{N,1}^i & f_{N,2}^i & \cdots & f_{N,k}^i & \cdots & f_{N,L}^i \end{bmatrix} \tag{2}$$

It can be seen from Eq. 1 and Eq. 2 that $\boldsymbol{X_i}$ is the statistical class feature of the ith station scene, and x_j^i is the jth of the total M statistical class features of the ith LVDN scene. $\boldsymbol{F_i}$ is the jth of a total of M statistical features of the ith LVDN scenario. Assuming that the scenario has N curve-like features and the maximum length of the data sequence is L, $\boldsymbol{F_i}$ is a matrix of the form N^*L, and $f_{j,k}^i$ is the kth data in the sequence of the jth curve-like feature of the ith LVDN scenario.

Then the above feature set is subjected to downscaling and clustering to achieve the classification of LVDN scenes and identify typical scenes. In this paper use PCA dimensionality reduction method to reduce the dimensionality of the data while retaining the main information. And through K-means clustering algorithm, the dataset is divided into groups with similar features, which represents a specific typical scene (Chen et al., 2019; Feng et al., 2020). The initial clustering centres are first selected for initialisation. Then the distance calculation is performed from each clustering centre is calculated. The expression is as follows:

$$d(x, c_i) = \sqrt{\sum_{j=1}^{m} \left(x_j - c_{ij}\right)^2} \tag{3}$$

In Eq. 3, x is the sample point, c_i is the ith clustering centre and m is the number of features of the sample.

Next assign the samples to the nearest cluster centres, forming K clusters. Then update the clustering centres. For each cluster, the mean of all samples is calculated and the mean is used as the new clustering centre. The updated clustering centre expression is:

$$c_i = \frac{1}{n_i} \sum_{x \in S_i} x \tag{4}$$

In Eq. 4, c_i is the ith cluster centre, n_i is the number of samples in the ith cluster, S_i is the set of samples in the ith cluster.

Finally, until the clustering centre no longer changes or the specified number of iterations is reached, the clustering results are analysed to identify different typical scenarios. Output the scene feature clustering results of LVDN. Based on the clustered features of the clustering centres, which are the feature inputs selected above. Analyse the differences in these features of each clustering centre to classify the different types of LVDN. The paper utilizes real data from LVDN. It contains the features of different LVDN and has wide applicability. The clustering method accurately classifies scenarios.

2.1.2 Application requirement identification of LVDN

The results of the analysis are used as labels, while the feature sets of the distribution network scenarios are used as inputs for building a decision tree-based model to determine the applications of LVDN (Sei et al., 2022; He et al., 2023). The Gain Ratio which is used to find the best partitioning attributes in decision tree algorithms can efficiently handle attributes with multiple values to find the correct application requirements for LVDN. The formula for the Gain Ratio is:

$$Gain\_ratio(D, a) = \frac{Gain(D, a)}{IV(a)} \tag{5}$$

$$IV(a) = -\sum_{i=1}^{V} \frac{|D^i|}{|D|} \log_2 \frac{|D^i|}{|D|} \tag{6}$$

In Eq. 5, $Gain\ (D,a)$ denotes the information gain obtained by segmentation using attribute A in dataset D. $IV\ (a)$ denotes the amount of information inherent in attribute A and is used as a measure of the diversity of values taken for attribute A.

2.2 Lightweight AI library construction

Lightweight AI library construction process mainly use python language environment, tensorflow framework, deep learning for algorithm model training. As well as tensorflow. lite on the algorithm lightweight processing, so that the algorithms can be better and more compatible to be deployed in LVDN.

To ensure compatibility with different hardware limitations and operating systems on various edge devices, tf.lite can be utilized. Firstly, tf.lite is optimized for diverse hardware platforms, including smartphones and embedded devices. It leverages hardware accelerators and specific instruction sets to enhance the execution efficiency of models. Secondly, tf.lite is compatible with multiple operating systems such as Android, iOS, and Linux. It provides interfaces and tools for interaction with these operating systems, simplifying the deployment and execution processes of models. tf.lite employs the FlatBuffer format for storing and transferring models. This format is compact and efficient, enabling fast cross-platform data transmission and parsing. During model loading, tf.lite converts the model into the FlatBuffer format.

According to the process mentioned above, the application of LVDN requirements are analysed. Then, the functional algorithms are developed to realise the application requirements. To adapt to limited computational resources at edge devices, the algorithmic modules are lightweighted. Techniques such as knowledge distillation training and model pruning are employed to obtain lightweight results.

The lightweight AI have smaller models, faster operation speeds, and can be deployed in a wide range of conditions compared to traditional AI deployed in LVDN. The lightweight AI library consists of three main module types:

1. Data processing module: This module handles data input, pre-processing, transformation, and output tasks. It includes algorithms for data cleaning, feature extraction, missing value filling, normalization.
2. Application function module: Designed to support specific application scenarios directly, such as PV output prediction, electric vehicle management, heavy overload prediction, and energy storage collaboration. These algorithms reduce computational complexity while maintaining high accuracy.
3. Maintenance management module: This module focuses on ensuring the continuous operation and maintenance of the algorithm library. It includes functions such as algorithm updates, anomaly detection, and fault recovery. This module enables the algorithm library to guarantee reliability.

3 Flexible orchestration of lightweight AI based on multi-factor priority

After the construction of the algorithm library is completed, the algorithm modules need to be flexibly embeded to each edge terminal within LVDN. The different orchestration of the algorithm modules will lead to the differentiation of the communication cost and operation efficiency among the edge terminals at runtime. The embedding scheme for algorithms is based on multi-factor priorities. Algorithms can be flexibly

TABLE 1 Typical service timing prioritization for LVDN.

Priority	Application requirement of low-voltage distribution network	Level of criticality
High priority	Relay protection, heavy overload prediction, fault monitoring, etc.,	3
Medium priority	Power trading, electricity dispatch, load frequency control, etc.,	2
Low priority	Electric energy metering, identification of abnormal electricity usage, load forecasting, etc.,	1

scheduled based on computational resources, operating conditions, and communication distance requirements. In this way, edge devices in LVDN can interact with each other, facilitating the flexible orchestration of algorithms in LVDN. Local processing reduces response time, data transmission costs and the risk of data loss, ensuring rapid and precise response to LVDN application requirements.

In this paper, we adopt the 0-1 variable $\theta^{i,j}$ to characterise the deployment situation between the algorithm module i and the edge computing terminal j, $j \in \{1, 2, \ldots, m\}$, where m is the number of edge computing terminals in the system, and the expression is:

$$\theta^{i,j} = \begin{cases} 1 \\ 0 \end{cases} \tag{7}$$

In Eq. 7, $\theta^{i,j}$ is 1, the algorithm module i is deployed to the edge computing terminal j. When $\theta^{i,j}$ is 0, the algorithm module i needs to match the remaining devices.

The objective function is to maximise the deployment priority of the algorithmic module and the expression is:

$$\max \sum \theta^{i,j} \cdot \eta_{i,j} \tag{8}$$

where $n_{i,j}$ represents the priority of algorithm module i deployed on edge computing terminal j.

3.1 Multi-factor prioritisation

In the multi-factor prioritization scheme, three types of factors are considered to align the algorithm deployment application with the current situation, including the criticality factor F_{im}, the communication distance factor F_{di}, and the computational resource factor F_{co}. The priority of each algorithm module embeded in different edge computing terminals can be expressed as follows:

$$\begin{aligned} \eta_{i,j} &= \lambda_1 \cdot F_{im} + \lambda_2 \cdot F_{di} + \lambda_3 \cdot F_{co} \\ s.t. \quad & 0 \le \lambda_1, \lambda_2, \lambda_3 \le 1 \\ & \lambda_1 + \lambda_2 + \lambda_3 = 1 \end{aligned} \tag{9}$$

It can be seen from Eq. 9 that $\lambda_1, \lambda_2, \lambda_3$ denote the control parameters of different factors which reflect the importance of variable factors. All control parameters are normalised to regulate within the range of variation [0,1] and sum to 1.

3.1.1 Criticality factor

In this paper, the services of LVDN are classified into three levels and quantitatively identified with corresponding numbers based on their timing priorities, as shown in Table 1.

High-priority operations demand an immediate response within seconds, while medium-priority operations allow for a response with a longer time compared to high-priority operations. Low-priority operations, such as statistical analysis of data, do not require an immediate response.

3.1.2 Communication distance factor

Since there is little difference in the cost of data interaction and power consumption between different communication methods, the communication distance factor F_{di} is a crucial element in determining the data transmission cost incurred during algorithm execution. F_{di} is calculated by normalizing the communication distance between side ends using the following expression:

$$F_{di} = \frac{e_{i,j} - \min\left(e_{i,j}\right)}{\max\left(e_{i,j}\right) - \min\left(e_{i,j}\right)} \; i = 1, 2, ..., n; \; j = 1, 2, ..., m \tag{10}$$

In Eq. 10, $e_{i,j}$ is the communication distance between the ith data node and the jth edge computing terminal; $min\ (e_{i,j})$ and $max\ (e_{i,j})$ denote the minimum and maximum values of the communication distance between edge ends.

3.1.3 Computational resource factor

To balance the computational resource variability among the edge terminals in LVDN, to prevent an edge devices from deploying many algorithmic modules, which leads to the full load of the device and reduces the lifetime of the device. Therefore the computational resource factor F_{co} is:

$$F_{co} = \begin{cases} 1, \dfrac{c_{re,j} + c_k}{c_{\max,j}} < P_j \\ 0, \dfrac{c_{re,j} + c_k}{c_{\max,j}} \ge P_j \end{cases} \tag{11}$$

It can be seen from Eq. 11 that $c_{re,j}$, $c_{max,j}$ are the occupied memory and the upper memory limit of the jth edge computing terminal; c_k is the memory required for the deployment of the algorithm module; and P_j is the memory warning value of the jth edge computing terminal.

3.2 Optimising processes

The overall optimisation process is shown in Figure 2. Firstly follow the above process data processing process and model construction process. Finally, using the particle swarm optimisation algorithm to solve the optimal deployment model. This algorithm is known for searching extensively in the solution space to find a globally optimal solution. The particle swarm optimisation algorithm can usually achive better results within fewer iterations and perform well in solving continuous optimisation problems.

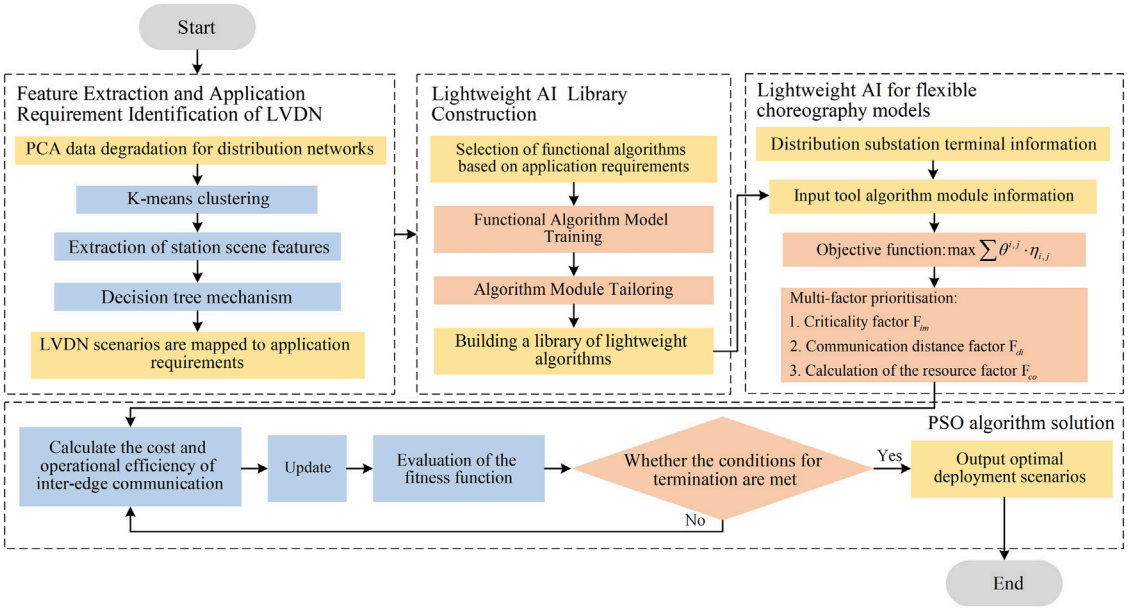

FIGURE 2
The overall optimisation process.

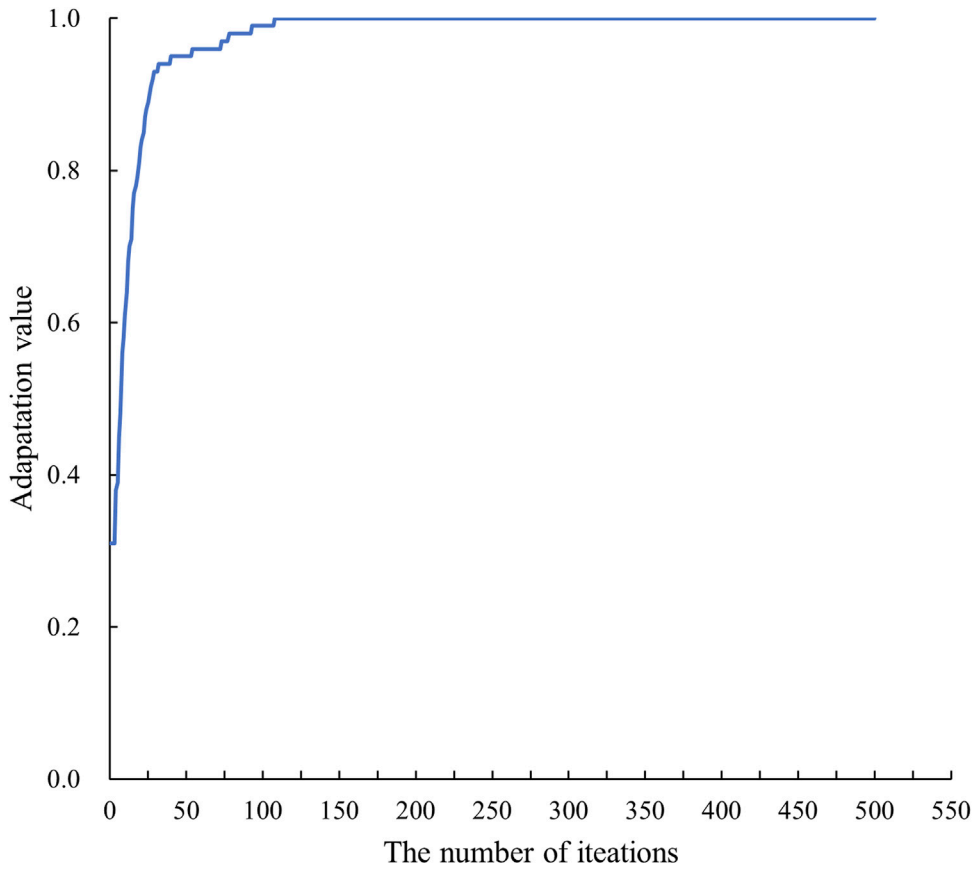

FIGURE 3
Adaptation curve.

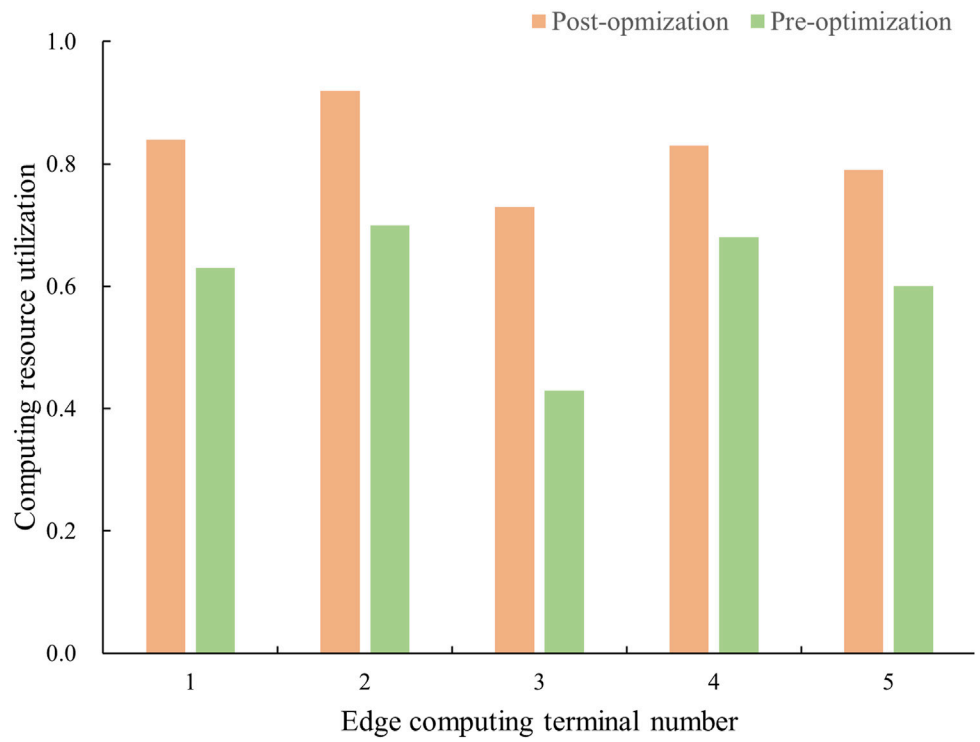

FIGURE 4
Computing resource utilization curve.

TABLE 2 Impact of different processing modes on the average processing time of operations.

	Average processing time of high-priority operations	Average processing time of medium-priority operations	Average processing time of low-priority operations
Methodology of this paper	0.37	0.93	1.62
Traditional model	0.76	1.42	2.13
Local processing	0.61	1.22	1.81

4 Case studies

In this paper, a 17-node LVDN in a region of Hangzhou, Zhejiang Province is taken as an example for simulation and analysis. Among them, nodes 1 to 5 are smart meters, nodes 6 to 12 are load switches, nodes 12 and 13 are energy storage devices, nodes 14 and 15 are distributed photovoltaic power sources, and nodes 16 and 17 are charging piles.

Based on the historical operation data spanning from 2022 to 2023, application requirements were identified. Subsequently, lightweight tools and algorithm libraries were developed to align with these application requirements. The optimisation model is then constructed by Matlab and solved by the particle swarm optimisation algorithm. Inalgorithm. In the optimisation search process, to better find the optimal algorithm module deployment scheme, the local and global learning factor in the particle swarm optimisation algorithm is set to 1.2431, the inertia weights are set to 0.8, and the number of iterations is 500.

In the optimization process, the inertia weights play a crucial role in speeding up the convergence of the particle swarm optimization algorithm, locating both local and global hyperparameter extremes efficiently. The learning factor regulates the acceleration relationship between local and global extremes to search for optimal parameters. As the adaptation value gradually converges and approaches stability, the search for the optimum configuration is completed. The evolution of the adaptation value can be visualized in Figure 3.

The deployment scenarios of different algorithmic modules have a significant impact on the computational resource utilization and the quality of services provided by each terminal. An increase of 26.4% in the average computing resource utilization rate of each terminal after optimization, as depicted in Figure 4.

Table 2 displays the average processing time of each priority service across various processing modes in the LVDN. Notably, high-priority operations exhibited optimal optimization, showcasing a significant reduction of 51.3 percent when compared to the traditional cloud computing model and a 39.3 percent decrease in comparison to the local processing model.

5 Conclusion

This research focuses on utilizing lightweight AI techniques for the flexible orchestration of algorithmic models through flexible collaboration. By processing LVDN tasks locally instead of uploading them to the cloud, the response time is reduced. This approach minimizes data transmission costs and the risk of data loss, while enabling a rapid and precise response to LVDN application requirements. Enhancing the efficiency and service quality of algorithmic model operations on terminals. Experimental results demonstrate an average reduction of 0.43 s in the optimized service response time and an average increase of 26.4% in edge computing terminal computing utilization.

This research is suitable for improving the efficiency of power service processing. However, it overlooks the aspects of security and confidentiality performance during service transmission and processing, which needs further improvement.

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Author contributions

YF: Writing–original draft. HW: Writing–original draft. ZL: Writing–original draft. JL: Writing–original draft. LL: Writing–original draft. XH: Writing–original draft. WC: Writing–original draft. BC: Writing–original draft.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This research was funded by Electric Power Research Institute of Fujian Electric Power Co., Ltd. (Research on the research of Lightweight AI development and multi-scenario applications for edge-side collaboration of low-voltage distribution network, Grant No. 52130422003N). The funder was not involved in the study design, collection, analysis, interpretation of data, the writing of this article, or the decision to submit it for publication.

Conflict of interest

Authors YF, HW, ZL, JL, LL, XH, WC, and BC were employed by Electric Power Research Institute of Fujian Electric Power Co., Ltd.

Publisher's note

References

Cai, T., Yao, H., Yang, Y., Zhang, Z., Ji, H., and Li, P. (2023). Cloud-edge collaboration-based supply restoration intelligent decision-making method. *Power Syst. Prot. Control* 51 (19), 94–103. doi:10.19783/j.cnki.pspc.221918

Chamola, V., Sancheti, A., Chakravarty, S., Kumar, N., and Guizani, M. (2020). An IoT and edge computing based framework for charge scheduling and EV selection in V2G systems. *IEEE Trans. Veh. Technol.* 69 (10), 10569–10580. doi:10.1109/TVT.2020.3013198

Chen, S., Wen, H., Wu, J., Lei, W., Hou, W., Liu, W., et al. (2019a). Internet of things based smart grids supported by intelligent edge computing. *IEEE Access* 7, 74089–74102. doi:10.1109/ACCESS.2019.2920488

Chen, X., Li, Z., Chen, Y., and Wang, X. (2019b). Performance analysis and uplink scheduling for QoS-aware NB-IoT networks in mobile computing. *IEEE Access* 7, 44404–44415. doi:10.1109/ACCESS.2019.2908985

Chen, X., Wen, H., Ni, W., Zhang, S., Wang, X., Xu, S., et al. (2022). Distributed online optimization of edge computing with mixed power supply of renewable energy and smart grid. *IEEE Trans. Commun.* 70 (1), 389–403. doi:10.1109/TCOMM.2021.3123275

Feng, Y., Zhao, S., and Liu, H. (2020). Analysis of network coverage optimization based on feedback K-means clustering and artificial fish swarm algorithm. *IEEE Access* 8, 42864–42876. doi:10.1109/ACCESS.2020.2970208

Fu, W., Wan, Y., Qin, J., Kang, Y., and Li, L. (2022). Privacy-preserving optimal energy management for smart grid with cloud-edge computing. *IEEE Trans. Industrial Inf.* 18 (6), 4029–4038. doi:10.1109/TII.2021.3114513

Han, J., Liu, N., and Shi, J. (2022). Optimal scheduling of distribution system with edge computing and data-driven modeling of demand response. *J. Mod. Power Syst. Clean Energy* 10 (4), 989–999. doi:10.35833/MPCE.2020.000510

He, X., Yu, T., Shen, Y., and Wang, S. (2023). Traffic processing model of big data base station based on hybrid improved CNN algorithm and K-centroids clustering algorithm. *IEEE Access* 11, 63057–63068. doi:10.1109/ACCESS.2023.3286860

Huo, W., Liu, F., Wang, L., Jin, Y., and Wang, L. (2020). Research on distributed power distribution fault detection based on edge computing. *IEEE Access* 8, 24643–24652. doi:10.1109/ACCESS.2019.2962176

Li, J., Gu, C., Xiang, Y., and Li, F. (2022). Edge-cloud computing systems for smart grid: state-of-the-art, architecture, and applications. *J. Mod. Power Syst. Clean Energy* 10 (4), 805–817. doi:10.35833/MPCE.2021.000161

Mudassar, M., Zhai, Y., and Lejian, L. (2022). Adaptive Fault-tolerant strategy for latency-aware IoT application executing in edge computing environment. *IEEE Internet Things J.* 9 (15), 13250–13262. doi:10.1109/JIOT.2022.3144026

Peng, N., Liang, R., Wang, G., Sun, P., Chen, C., and Hou, T. (2021). Edge computing-based fault location in distribution networks by using asynchronous transient amplitudes at limited nodes. *IEEE Trans. Smart Grid* 12 (1), 574–588. doi:10.1109/TSG.2020.3009005

Perin, G., Berno, M., Erseghe, T., and Rossi, M. (2022). Towards sustainable edge computing through renewable energy resources and online, distributed and predictive scheduling. *IEEE Trans. Netw. Serv. Manag.* 19 (1), 306–321. doi:10.1109/TNSM.2021.3112796

Sei, Y., Onesimu, J. A., and Ohsuga, A. (2022). Machine learning model generation with copula-based synthetic dataset for local differentially private numerical data. *IEEE Access* 10, 101656–101671. doi:10.1109/ACCESS.2022.3208715

Wang, J., and Peng, Y. (2020). Distributed optimal dispatching of multi-entity distribution network with demand response and edge computing. *IEEE Access* 8, 141923–141931. doi:10.1109/ACCESS.2020.3013231

Wang, K., Wu, J., Zheng, X., Li, J., Yang, W., and Vasilakos, A. V. (2023). Cloud-edge orchestrated power dispatching for smart grid with distributed energy resources. *IEEE Trans. Cloud Comput.* 11 (2), 1194–1203. doi:10.1109/TCC.2022.3185170

Yue, D., He, Z., and Dou, C. (2023). Cloud-edge collaboration-based distribution network reconfiguration for voltage preventive control. *IEEE Trans. Industrial Inf.* 19 (12), 11542–11552. doi:10.1109/TII.2023.3247028

Zhong, J., Liu, B., Yu, X., Wong, P., Wang, Z., Xu, C., et al. (2023). Enhancing voltage compliance in distribution network under cloud and edge computing framework. *IEEE Trans. Cloud Comput.* 11 (2), 1217–1229. doi:10.1109/TCC.2022.3149238